US-PAT-NO:	6785765
DOCUMENT-IDENTIFIER:	US 6785765 B1
TITLE:	Status register to improve initialization of a synchronous memory

---------- KWIC ---------

Detailed Description Text - DETX (13):

Bank address input connections, BA0 and BA1 define which bank an ACTIVE, READ, WRITE, or BLOCK PROTECT command is being applied. The DQ0-DQ15 connections 143 are data bus connections used for bi-directional data communication. Referring to FIG. 1B, a VCCQ connection is used to provide isolated power to the DQ connections to improved noise immunity. In one embodiment, VCCQ=**Vcc** or 1.8V .+-.0.15V. The VSSQ connection is used to isolated ground to DQs for improved noise immunity. The **VCC** connection provides a power supply, such as 3V. A ground connection is provided through the Vss connection. Another optional voltage is provided on the VCCP connection 144. The VCCP connection can be tied externally to **VCC,** and sources current during device initialization, WRITE and ERASE operations. That is, writing or erasing to the memory device can be performed using a VCCP voltage, while all other operations can be performed with a **VCC** voltage. The Vccp connection is coupled to a high voltage switch/pump circuit 145.

Detailed Description Text - DETX (18):

In general, the synchronous flash memory is configured similar to a multi-bank DRAM that operates at low voltage and includes a synchronous interface. Each of the banks is organized into rows and columns. Prior to normal operation, the synchronous **flash memory is initialized**. The following sections provide detailed information covering device initialization, register definition, command descriptions and device operation.

Detailed Description Text - DETX (19):

The synchronous flash is powered up and initialized in a predefined manner. After power is applied to VCC, VCCQ and VCCP (simultaneously), and the clock signal is stable, RP# 140 is brought from a LOW state to a HIGH state. A delay, such as a 100 .mu.s delay, is needed after RP# transitions HIGH in order to complete internal device initialization. After the delay time has passed, the memory is placed in an array read mode and is ready for Mode Register programming or an executable command. After initial programming of a non-volatile mode register 147 (NVMode Register), the contents are automatically loaded into a volatile Mode Register 148 during the initialization. The device will power up in a programmed state and will not require reloading of the non-volatile mode register 147 prior to issuing operational commands. This is explained in greater detail below.

Detailed Description Text - DETX (30):

To allow for maximum power conservation, the synchronous flash features a very low current, deep power-down mode. To enter this mode, the RP# pin 140 (reset/power-down) is taken to VSS.+-.0.2V. To prevent an inadvertent RESET, RP# must be held at Vss for 100 ns prior to the device entering the reset mode. With RP# held at Vss, the device will enter the deep power-down mode. After the device enters the deep power-down mode, a transition from LOW to HIGH on RP# will result in a device power-up initialize sequence as outlined herein. Transitioning RP# from LOW to HIGH after entering the reset mode but prior to entering deep power-down mode requires a 1 .mu.s delay prior to issuing an executable command. When the device enters the deep power-down mode, all buffers excluding the RP# buffer are disabled and the current draw is low, for example, a maximum of 50 .mu.A at 3.3V VCC. The input to RP# must remain at Vss during deep power-down. Entering the RESET mode clears the Status Register 134 and sets the ISM 132 to the array read mode.

Detailed Description Text - DETX (36):

Upon power-up and prior to issuing any operational commands to the device, the synchronous flash is initialized. After power is applied to **VCC,** VCCQ and VCCP (simultaneously), and the clock is stable, RP# is transitioned from LOW to HIGH. A delay (in one embodiment a 100 .mu.s delay) is required after RP# transitions HIGH in order to complete internal device initialization. The device is in the array read mode at the completion of device initialization, and an executable command can be issued to the device.

Detailed Description Text - DETX (54):

In prior memories, a **Vcc** detector circuit could be used that would signal if the power applied to the device was high enough to read the register. Once the power was sufficient, then the circuitry could start to read these fuse elements and load them into some volatile registers. These **Vcc** detector circuits are not reliable such that the level that they detect varies widely. In a typical case, these detectors could change from a detection level of 1.4v to 2.7v depending on process and temperature variations. Thus, reading of the fuses needed to be done at a worst-case situation, and the circuitry is very complicated and large because reading Flash cells with 1.4v is difficult.

Detailed Description Text - DETX (56):

In one embodiment, the present invention requires the memory controller of the system to issue a device initialize command to the memory. The Flash then would start reading data from fuse elements. Since the **Vcc** is good at that time, the device uses the normal circuits on the chip to do the data reads and is smaller and more effective. In one embodiment, therefore, the present invention receives an initialize command that indicates that **Vcc** is at an appropriate level. The memory then reads the non-volatile "fuses".

Detailed Description Text - DETX (57):

In another embodiment, the **Flash memory** uses the reset connection (RP#) to perform an **initialization** process. The RP connection is used for multiple functions in the memory. One function is to allow access to protection circuitry in the memory. By providing an elevated voltage on the RP connection, the device protection modes can be ignored, and the memory protect register can be edited. The RP connection is also used to reset the device on power-up. Here the RP pin receives a voltage that transitions to a de-asserted (high) state when the memory is powered up. The memory then begins an initialization process, including reading the fuses. The memory includes a "filter" that prevents an initialization sequence from being performed if the RP connection is not low for a predetermined time. For example, in one embodiment the RP connection must be low (Vss) for at least 100 ns prior to entering a reset/initialization operation. This time requirement prevents accidental reset due to noise on the RP connection.

Detailed Description Text - DETX (59):

The present invention, therefore, can also perform initialization operations based on the RP connection. Two different **initialization** schemes can be used with the present synchronous **Flash memory**. The first uses an initialize command from the memory controller of the system. The second uses the RP connection to begin initialization. This embodiment reduces the need for the system to provide a specific command, which may be difficult in some systems.

Detailed Description Text - DETX (62):

As described above, the initialization operation can be started via hardware or software once **Vcc** has reached a proper value. The initialization operation requires an undetermined amount of time, and memory operations cannot be performed until the initialization is done. One choice for the end user is to wait a specified period of time, such as 100 .mu.s, after power up and issuance of a proper command. This period of time is selected to be more time than required to perform the initialization operation. Thus, after this time period, the memory is initialized and ready. The initialization sequence, however, may not take that much time. In one embodiment, about 40 to 45 .mu.s is needed to finish an Initialization sequence.

US-PAT-NO: 6715067

DOCUMENT-IDENTIFIER: US 6715067 B1

TITLE: Initializing a processor-based system from a non-volatile re-programmable semiconductor memory

---------- KWIC ---------

Detailed Description Text - DETX (29):

Turning now to FIG. 7, in accordance with one embodiment, software that uses the FLAT to allow multiple code and data images to be stored in **FLASH** memory, begins on **power up** or system reset with the BIOS executing and performing system **initialization** and Power on Self Test activities (block 110). The contents of the FLASH memory may be validated by checking the CRC stored at field 96 in the FLASH memory, as indicated in block 112. At this point, the BIOS selects the boot loader (block 114) to execute by scanning the FLAT and selecting the entry marked as the boot loader. The boot loader then uses the FLAT to find where in the FLASH memory the primary operating system is located (block 116), loads the operating system at the appropriate address in system memory (block 118) and starts its execution (block 120).

Current US Original Classification - CCOR (1):

713/1

Current US Cross Reference Classification - CCXR (2):

713/2

US-PAT-NO: 6329852

DOCUMENT-IDENTIFIER: US 6329852 B1

TITLE: Power on reset circuit

---------- KWIC ---------

Brief Summary Text - BSTX (5):

A **flash** EEPROM generally includes logic circuits, and after a setup of a
power supply **voltage,** the logic circuits should be **initialized** with a
predetermined state. Thus, a power on reset circuit is used to generate a
signal capable of initializing the logic circuits as soon as the power supply
voltage is set up.

Current US Original Classification - CCOR (1):

327/143

US-PAT-NO: 5835935

DOCUMENT-IDENTIFIER: US 5835935 A

TITLE: Method of and architecture for controlling system data with automatic wear leveling in a semiconductor non-volatile mass storage memory

---------- KWIC ---------

Detailed Description Text - DETX (14):

To speed up the operation of the present invention, a block RAM is included and used to store a table representing the LBAs, corresponding PBAs and relevant data. FIG. 7 shows a schematic representation of the table 144 stored within the block RAM for the present invention. The block RAM table includes an entry 132 for each LBA in the system file partition plus some spare locations for increased performance. The entries for each LBA 0 through N are **initialized at power up** to a corresponding cluster PBA 0 through N by reading the LBA field 108 in the **flash** extension. An entry for each PBA of the spare clusters is also included within the table. Each entry 132 includes a latest physical block address of the cluster (PBA) field 134 of the information associated with an LBA as well as a mirror of the old flag 138, the used flag 140, and the defect flag 142 associated with the actual physical block address of the cluster in the semiconductor non-volatile mass storage memory. Also, the field 136 represents the most recent data or sector within the given physical block address associated with the given LBA. Therefore, if the block size (cluster size) is 8, and for example the most recent sector within this cluster is 5, the field 136 is loaded with the value 5. This will help the controller during an access to locate the most recent sector within the

cluster. Alternatively, this field can also be removed if the controller reads
every sector within the cluster to locate the most recent system file.

Current US Original Classification - CCOR (1):
711/103

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L2	19822	((711/103) or (711/167) or (711/154) or (711/166) or (711/170) or (365/203) or (365/226) or (365/201) or (365/228) or (713/1) or (713/2) or (712/37) or (710/10) or (710/35) or (710/14) or (714/718) or (714/721) or (327/142) or (327/143)).CCLS.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2004/12/10 12:59
L3	1513	2 and flash and initializ$5	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 13:00
L4	320	2 and (flash with initializ$5)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 13:04
L5	19	2 and (flash adj initializ$5)	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 13:00
L6	65	2 and ((flash with initializ$5) with (power-up or powerup or (power adj up) or vcc or voltage))	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 13:06
S1	15	(("5416363") or ("5581206") or ("5615159") or ("5677885") or ("5717639") or ("5723990") or ("5767711") or ("5887162") or ("5896551") or ("5905909") or ("6046615") or ("6167495") or ("6178501") or ("6229352") or ("6246626")).PN.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2004/12/08 16:51
S2	32	(("5416363") or ("5581206") or ("5615159") or ("5677885") or ("5717639") or ("5723990") or ("5767711") or ("5887162") or ("5896551") or ("5905909") or ("6046615") or ("6167495") or ("6178501") or ("6229352") or ("6246626") or ("5440632") or ("6560161") or ("6654311") or ("6757211") or ("6314049") or ("6785764") or ("6675255") or ("6697907") or ("6728161") or ("6327202") or ("6570791") or ("6741497") or ("6101150") or ("6438068") or ("5617350") or ("5682345") or ("5864499")).PN.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2004/12/08 18:07

S3	20	S2 and synchronous	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:08
S4	14	S2 and synchronous and flash	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:08
S5	9	S2 and synchronous and flash and initializ$5	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:24
S6	3675	not S5 and synchronous and flash and memory and initializ$5	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:26
S7	476	not S5 and synchronous and flash and memory and initializ$5 and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:27
S8	62	not S5 and (synchronous adj flash) and memory and initializ$5 and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:42
S9	990	not S5 and (flash adj memory) and initializ$5 and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:42
S10	141	not S5 and ((flash adj memory) with initializ$5) and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:46
S11	1	not S5 and ((flash adj memory) with initializ$5) with vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:43
S12	0	("2004/0199713").URPN.	USPAT	OR	ON	2004/12/08 18:45
S13	141	not S5 and ((flash adj memory) with initializ$5) and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/08 18:47
S14	0	("2004/0199713").URPN.	USPAT	OR	ON	2004/12/10 08:40

S15	32	(("5416363") or ("5581206") or ("5615159") or ("5677885") or ("5717639") or ("5723990") or ("5767711") or ("5887162") or ("5896551") or ("5905909") or ("6046615") or ("6167495") or ("6178501") or ("6229352") or ("6246626") or ("5440632") or ("6560161") or ("6654311") or ("6757211") or ("6314049") or ("6785764") or ("6675255") or ("6697907") or ("6728161") or ("6327202") or ("6570791") or ("6741497") or ("6101150") or ("6438068") or ("5617350") or ("5682345") or ("5864499")).PN.	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	OFF	2004/12/10 08:41
S16	9	S15 and synchronous and flash and initializ$5	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 08:41
S17	141	not S16 and ((flash adj memory) with initializ$5) and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 08:47
S18	9	S15 and synchronous and flash and initializ$5	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 08:47
S19	148	((flash adj memory) with initializ$5) and vcc	US-PGPUB; USPAT; EPO; JPO; IBM_TDB	OR	ON	2004/12/10 12:58

About | Products | Investors | Sales | Jobs | News

DRAM | Modules | Flash | PSRAM | CMOS Image Sensors

Parametric Part

My Profile My Pa

Home > Search

Content Search

Parametric Part Search | Part Number Search | Content Search

Search Results

initialize and flash memory Search!

Your query "**initialize and flash memory**" matched 12 documents.

1 2 [Next]

83% - Technical Data Sheet

Summary: 2 MEG x 16 ASYNC/ PAGE FLASH MEMORY 09005aef808cfe29 Micron Technology, Inc., reserves the right to change specifications without notice. Command State Machine Operations The CSM decodes instructions for read array, read protection

82% - 1Q01 BN

Summary: This product introduction confirms Micron's commitment to provide our customers with a 100 percent compatible alt for Intel's StrataFlash « device. Product and service news for Micron customers 2 mbn extra: (continued from page 1) Q- Flas...

82% - Technical Data Sheet

Summary: TN- 28- 72 QUALCOMM EFS SOFTWARE 5000/ 5100 PLATFORM TECHNICAL NOTE USING MICRON FLASH WITH QUA SOFTWARE FOR 5000/ 5100 PLATFORM Introduction Assumptions made in the EFS software are incorrect when using Micron « N W30 low- power Flas...

82% - 128Mb, 64Mb, 32Mb Q-Flash Memory

Summary: PRODUCTS ARE ONLY WARRANTED BY MICRON TO MEET MICRON'S PRODUCTION DATA SHEET SPECIFICATIONS. 1 32Mb Q- FLASH MEMORY 09005aef80b5a323 Micron Technology, Inc., reserves the right to change products or specifications wi Table ...

82% - 128Mb, 64Mb, 32Mb Q-Flash Memory

Summary: PRODUCTS ARE ONLY WARRANTED BY MICRON TO MEET MICRON'S PRODUCTION DATA SHEET SPECIFICATIONS. 1 32Mb Q- FLASH MEMORY 09005aef80b5a323 Micron Technology, Inc., reserves the right to change products or specifications wi Table ...

82% - 128Mb, 64Mb, 32Mb Q-Flash Memory

Summary: M 10/ 04 EN 1 ©2000 Micron Technology, Inc. 256Mb, 128Mb, 64Mb, 32Mb Q- FLASH MEMORY Q- FLASH MEMORY I MT28F128J3, MT28F640J3, MT28F320J3 Features Memory Organization ° x8/ x16 ° Two hundred fifty- six 128KB erase blocks (: hundre...

80% - Technical Data Sheet

Summary: REPLACING INTEL 28F128W18 or 28F128L18 WITH MICRON MT28F1284W18 TECHNICAL NOTE FDI SOFTWARE CON REPLACING INTEL . REPLACING INTEL 28F128W18 OR 28F128L18 DEVICES WITH THE MICRON MT28F1284W18 DEVICE Modifyi Section Overview FDI su...

80% - MT28F1284W18.book

Summary: 8 MEG x 16 ASYNC/ PAGE/ BURST FLASH MEMORY FLASH MEMORY MT28F1284W18 1.8V Low Voltage, Extended Te Features Dedicated commands to decrease programming times for both in- factory and in- system operations Fast programming for fast...

80% - MT28F644W30.book.fm

Summary: 4 MEG x 16 ASYNC/ PAGE/ BURST FLASH MEMORY FLASH MEMORY MT28F644W18 MT28F644W30 1.8V Low Voltage Temperature Features ° Flexible 4Mb multipartition architecture ° Single word (16- bit) data bus ° Support for true concurrent o zero ...

77% - 4 Meg x 16 Async/Page/Burst CellularRAM 1.0 Memory

Summary: 09005aef80be1fbd pdf/ 09005aef80be2036 zip Burst CellularRAM_ 1. fm - Rev. 4 MEG x 16 ASYNC/ PAGE/ BURST C MEMORY ASYNC/ PAGE/ BURST CellularRAM TM 1.0 MEMORY MT45W4MW16BFB Features ° Single device supports asynchronous burst op...

1 2 [Next]

Legal | Privacy Policy | Contact

Micron

About | Products | Investors | Sales | Jobs | News

DRAM | Modules | Flash | PSRAM | CMOS Image Sensors

Parametric Part

Home > Search

My Profile My Pa

Content Search

Parametric Part Search | Part Number Search | Content Search

Search Results

no RP# Search!

Your query "**no RP#**" matched 0 documents.

Legal | Privacy Policy | Contact

Micron

About | Products | Investors | Sales | Jobs | News

DRAM | Modules | Flash | PSRAM | CMOS Image Sensors

Parametric Part

Home > Search

My Profile My Pa

Content Search

Parametric Part Search | Part Number Search | Content Search

Search Results

synchronous flash initialization Search!

Your query "**synchronous flash initialization**" matched 0 documents.

Legal | Privacy Policy | Contact

Micron

About | Products | Investors | Sales | Jobs | News

DRAM | Modules | Flash | PSRAM | CMOS Image Sensors

Parametric Part

Home > Search

My Profile My Pa

Content Search

Parametric Part Search | Part Number Search | Content Search

Search Results

synchronous flash Search!

Your query "**synchronous flash** " matched 0 documents.

Legal | Privacy Policy | Contact

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+synchronous +flash +memory +initialize SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **synchronous flash memory initialize**

Found **41** of **147,060**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 41

Result page: **1** 2 3 next

Relevance scale □ ▄ ▄ ■ ■

1 Soft updates: a solution to the metadata update problem in file systems

Gregory R. Ganger, Marshall Kirk McKusick, Craig A. N. Soules, Yale N. Patt

May 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 2

Full text available: pdf(147.90 KB)

Additional Information: full citation, abstract, references, citings, index terms

Metadata updates, such as file creation and block allocation, have consistently been identified as a source of performance, integrity, security, and availability problems for file systems. Soft updates is an implementation technique for low-cost sequencing of fine-grained updates to write-back cache blocks. Using soft updates to track and enforce metadata update dependencies, a file system can safely use delayed writes for almost all file operations. This article describes soft ...

2 Integration of message passing and shared memory in the Stanford FLASH multiprocessor

John Heinlein, Kourosh Gharachorloo, Scott Dresser, Anoop Gupta

November 1994 **Proceedings of the sixth international conference on Architectural support for programming languages and operating systems**, Volume 29 , 28 Issue 11 , 5

Full text available: pdf(1.80 MB)

Additional Information: full citation, abstract, references, citings, index terms

The advantages of using message passing over shared memory for certain types of communication and synchronization have provided an incentive to integrate both models within a single architecture. A key goal of the FLASH (FLexible Architecture for SHared memory) project at Stanford is to achieve this integration while maintaining a simple and efficient design. This paper presents the hardware and software mechanisms in FLASH to support various message passing protocols. We achieve low overhe ...

3 Where is time spent in message-passing and shared-memory programs?

Satish Chandra, James R. Larus, Anne Rogers

November 1994 **Proceedings of the sixth international conference on Architectural support for programming languages and operating systems**, Volume 29 , 28 Issue 11 , 5

Full text available: pdf(1.55 MB)

Additional Information: full citation, abstract, references, citings, index terms

Message passing and shared memory are two techniques parallel programs use for coordination and communication. This paper studies the strengths and weaknesses of these two mechanisms by comparing equivalent, well-written message-passing and shared-memory programs running on similar hardware. To ensure that our measurements are comparable, we produced two carefully tuned versions of each program and measured them on closely-related simulators of a message-passing and a shared-memory machine, ...

4 The Rio file cache: surviving operating system crashes
Peter M. Chen, Wee Teck Ng, Subhachandra Chandra, Christopher Aycock, Gurushankar Rajamani, David Lowell
September 1996 **Proceedings of the seventh international conference on Architectural support for programming languages and operating systems**, Volume 31 , 30 Issue 9 , 5
Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings, index terms

One of the fundamental limits to high-performance, high-reliability file systems is memory's vulnerability to system crashes. Because memory is viewed as unsafe, systems periodically write data back to disk. The extra disk traffic lowers performance, and the delay period before data is safe lowers reliability. The goal of the Rio (RAM I/O) file cache is to make ordinary main memory safe for persistent storage by enabling memory to survive operating system crashes. Reliable memory enables a syste ...

5 Session 13: scheduling and operating systems: Application-specific protocols for user-level shared memory
Babak Falsafi, Alvin R. Lebeck, Steven K. Reinhardt, Ioannis Schoinas, Mark D. Hill, James R. Larus, Anne Rogers, David A. Wood
November 1994 **Proceedings of the 1994 ACM/IEEE conference on Supercomputing**
Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings

Recent distributed shared memory (DSM) systems and proposed shared-memory machines have implemented some or all of their cache coherence protocols in software. One way to exploit the flexibility of this software is to tailor a coherence protocol to match an application's communication patterns and memory semantics. This paper presents evidence that this approach can lead to large performance improvements. It shows that application-specific protocols substantially improved the performance of t ...

6 Integrating reliable memory in databases
Wee Teck Ng, Peter M. Chen
August 1998 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 7 Issue 3
Full text available: pdf(123.18 KB) Additional Information: full citation, abstract, index terms

Recent results in the Rio project at the University of Michigan show that it is possible to create an area of main memory that is as safe as disk from operating system crashes. This paper explores how to integrate the reliable memory provided by the Rio file cache into a database system. Prior studies have analyzed the performance benefits of reliable memory; we focus instead on how different designs affect reliability. We propose three designs for integrating reliable memory into databases: non ...

Keywords: Main memory database system (MMDB), Recovery, Reliability

7 ESP: a language for programmable devices
Sanjeev Kumar, Yitzhak Mandelbaum, Xiang Yu, Kai Li
May 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2001 conference on Programming language design and implementation**, Volume 36 Issue 5
Full text available: pdf(1.60 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents the design and implementation of Event-driven State-machines Programming (ESP)—a language for programmable devices. In traditional languages, like C, using event-driven state-machine forces a tradeoff that requires giving up ease of development and reliability to achieve high performance. ESP is designed to provide all of these three properties simultaneously.

ESP provides a comprehensive set of features to support development of compact and modular programs. ...

8 Efficient synchronization: let them eat QOLB
Alain Kägi, Doug Burger, James R. Goodman
May 1997 **ACM SIGARCH Computer Architecture News , Proceedings of the 24th annual international symposium on Computer architecture**, Volume 25 Issue 2
Full text available: pdf(2.04 MB) Additional Information: full citation, abstract, references, citings, index terms

Efficient synchronization primitives are essential for achieving high performance in fine-grain, shared-memory parallel programs. One function of synchronization primitives is to enable exclusive access to shared data and critical sections of code. This paper makes three contributions. (1) We enumerate the five sources of overhead that locking synchronization primitives can incur. (2) We describe four mechanisms (local spinning, queue-based locking, collocation, and synchronized prefetch) that r ...

9 Building reliable mobile-aware applications using the Rover toolkit
Anthony D. Joseph, M. Frans Kaashoek
October 1997 **Wireless Networks**, Volume 3 Issue 5
Full text available: pdf(371.04 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses extensions to the Rover toolkit for constructing reliable mobile-aware applications. The extensions improve upon the existing failure model, which addresses client or communication failures and guarantees reliable message delivery from clients to server, but does not address server failures (e.g., the loss of an incoming message due to server failure) (Joseph et al., 1997). Due to the unpredictable, intermittent communication connectivity typically found in mobile clien ...

10 Performance experiences on Sun's Wildfire prototype
Lisa Noordergraaf, Ruud van der Pas
January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM)**
Full text available: pdf(248.16 KB) Additional Information: full citation, references, citings, index terms

11 Location-awareness and interworking: Proximity services supporting network virtual memory in mobile devices
Emanuele Lattanzi, Andrea Acquaviva, Alessandro Bogliolo
October 2004 **Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots**
Full text available: pdf(192.89 KB) Additional Information: full citation, abstract, references, index terms

Wireless networked embedded terminals like personal digital assistants, cell-phones or sensor nodes are typically memory constrained devices. This limitation prevents the development of applications that require a large amount of run-time memory space. In a wired cum wireless scenario, a potentially unlimited amount of virtual memory can be found on remote servers installed on the wired network. However, virtual memory access requires performance constrained and lossless data flows against te ...

Keywords: mobility management, network swapping, proximity service, wireless networks

12 Cashmere-2L: software coherent shared memory on a clustered remote-write network
Robert Stets, Sandhya Dwarkadas, Nikolaos Hardavellas, Galen Hunt, Leonidas Kontothanassis, Srinivasan Parthasarathy, Michael Scott
October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles**, Volume 31 Issue 5
Full text available: pdf(2.17 MB) Additional Information: full citation, references, citings, index terms

13 Building reliable mobile-aware applications using the Rover toolkit
Anthony D. Joseph, M. Frans Kaashoek
November 1996 **Proceedings of the 2nd annual international conference on Mobile computing and networking**
Full text available: pdf(1.36 MB) Additional Information: full citation, references, citings, index terms

14 Robustness: Using model checking to debug device firmware
Sanjeev Kumar, Kai Li
December 2002 **ACM SIGOPS Operating Systems Review**, Volume 36 Issue SI
Full text available: pdf(1.72 MB) Additional Information: full citation, abstract, references

Device firmware is a piece of concurrent software that achieves high performance at the cost of software complexity. They contain subtle race conditions that make them difficult to debug using traditional debugging techniques. The problem is further compounded by the lack of debugging support on the devices. This is a serious problem because the device firmware is trusted by the operating system.Model checkers are designed to systematically verify properties of concurrent systems. Therefore, mod ...

15 A distributed 3D graphics library
Blair MacIntyre, Steven Feiner
July 1998 **Proceedings of the 25th annual conference on Computer graphics and interactive techniques**
Full text available: pdf(355.83 KB) Additional Information: full citation, references, citings, index terms

Keywords: distributed shared memory, distributed virtual environments, object-oriented graphics, shared-data object model

16 Remote queues: exposing message queues for optimization and atomicity
Eric A. Brewer, Frederic T. Chong, Lok T. Liu, Shamik D. Sharma, John D. Kubiatowicz
July 1995 **Proceedings of the seventh annual ACM symposium on Parallel algorithms and architectures**
Full text available: pdf(1.78 MB) Additional Information: full citation, references, citings, index terms

17 The impact of architectural trends on operating system performance
M. Rosenblum, E. Bugnion, S. A. Herrod, E. Witchel, A. Gupta

December 1995 **ACM SIGOPS Operating Systems Review , Proceedings of the fifteenth ACM symposium on Operating systems principles**, Volume 29 Issue 5
Full text available: pdf(2.03 MB) Additional Information: full citation, references, citings, index terms

18 Features: The Inevitability of Reconfigurable Systems
Nick Tredennick, Brion Shimamoto
October 2003 **Queue**, Volume 1 Issue 7
Full text available: pdf(1.64 MB) html(40.12 KB) Additional Information: full citation, index terms

19 Static analysis to reduce synchronization costs in data-parallel programs
Manish Gupta, Edith Schonberg
January 1996 **Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages**
Full text available: pdf(1.14 MB) Additional Information: full citation, references, citings, index terms

20 The *nesC* language: A holistic approach to networked embedded systems
David Gay, Philip Levis, Robert von Behren, Matt Welsh, Eric Brewer, David Culler
May 2003 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2003 conference on Programming language design and implementation**, Volume 38 Issue 5
Full text available: pdf(177.98 KB) Additional Information: full citation, abstract, references, citings, index terms

We present *nesC*, a programming language for networked embedded systems that represent a new design space for application developers. An example of a networked embedded system is a sensor network, which consists of (potentially) thousands of tiny, low-power "motes," each of which execute concurrent, reactive programs that must operate with severe memory and power constraints.nesC's contribution is to support the special needs of this domain by exposing a programming model that incorporates ...

Keywords: C, TinyOS, components, concurrency, data races, first-order, modules, nesC, programming languages

Results 1 - 20 of 41 Result page: **1** 2 3 next

PORTAL
US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+synchronous +"flash memory" +initialize SEARCH

Feedback Report a problem Satisfaction survey

Terms used **synchronous** **flash memory** **initialize**

Found **8** of **147,060**

Sort results by: relevance
Display results: expanded form

Save results to a Binder
Search Tips
□ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 8 of 8

Relevance scale

1 Verification: Model based estimation and verification of mobile device performance
Gopal Raghavan, Ari Salomaki, Raimondas Lencevicius
September 2004 **Proceedings of the fourth ACM international conference on Embedded software**
Full text available: pdf(252.67 KB) Additional Information: full citation, abstract, references, index terms

Performance is an important quality attribute that needs to be planned and managed proactively. Abstract models of the system are not very useful if they do not produce reasonably accurate metrics. Detailed models are time consuming and expensive to build as well as to simulate. In order to strike a right balance, a framework is proposed in this paper that takes advantage of the flexibility of abstract modeling and intricacies of detailed modeling. Performance is modeled and verified per use cas ...

Keywords: performance analysis, system level modeling, use case verification

2 Building reliable mobile-aware applications using the Rover toolkit
Anthony D. Joseph, M. Frans Kaashoek
October 1997 **Wireless Networks**, Volume 3 Issue 5
Full text available: pdf(371.04 KB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses extensions to the Rover toolkit for constructing reliable mobile-aware applications. The extensions improve upon the existing failure model, which addresses client or communication failures and guarantees reliable message delivery from clients to server, but does not address server failures (e.g., the loss of an incoming message due to server failure) (Joseph et al., 1997). Due to the unpredictable, intermittent communication connectivity typically found in mobile clien ...

3 Building reliable mobile-aware applications using the Rover toolkit
Anthony D. Joseph, M. Frans Kaashoek
November 1996 **Proceedings of the 2nd annual international conference on Mobile computing and networking**
Full text available: pdf(1.36 MB) Additional Information: full citation, references, citings, index terms

4 *Interactors*: a real-time executive with multiparty interactions in C++
Pierre Labrèche, Louis Lamarche

April 1990 **ACM SIGPLAN Notices**, Volume 25 Issue 4

Full text available: pdf(790.56 KB) Additional Information: full citation, abstract, references

Interactors is a run-time environment for embedded real-time software, which adds concurrency to the C++ object-oriented language. *Interactors* allows sequential processes to interact synchronously or asynchronously, and provides user-definable multiparty interactions. Several forms of selective wait, inspired by Ada, are provided. Scheduling algorithms follow Carnegie-Mellon University's recommendations for implementing hard deadline scheduling. Concepts are illustrated by simple a ...

5 Partial reconfigurable architectures: Real-time LUT-based network topologies for dynamic and partial FPGA self-reconfiguration

Michael Huebner, Tobias Becker, Juergen Becker

September 2004 **Proceedings of the 17th symposium on Integrated circuits and system design**

Full text available: pdf(356.70 KB) Additional Information: full citation, abstract, references, index terms

Xilinx Virtex FPGAs offer the possibility of dynamic and partial run-time reconfiguration. If a system uses this feature the designer has to take care, that no signal lines cross the border to other reconfigurable regions. Traditional solutions connecting modules on a dynamic and partial reconfigurable system use TBUF elements for connection and separation of the functional blocks. While automalically placing and routing the design, the routing-tool sometimes uses signal lines which cross the mo ...

Keywords: dynamic partial reconfiguration, virtex

6 Features: The Inevitability of Reconfigurable Systems

Nick Tredennick, Brion Shimamoto

October 2003 **Queue**, Volume 1 Issue 7

Full text available: pdf(1.64 MB) html(40.12 KB) Additional Information: full citation, index terms

7 Special issue on wireless pan & sensor networks: A study of energy consumption and reliability in a multi-hop sensor network

Jonathan M. Reason, Jan M. Rabaey

January 2004 **ACM SIGMOBILE Mobile Computing and Communications Review**, Volume 8 Issue 1

Full text available: pdf(477.91 KB) Additional Information: full citation, abstract, references

For a moderate-size, multi-hop, sensor network, we present experimental measurements of radio energy consumption and packet reliability. We categorize the energy measurements by energy consumed in each radio state and for each traffic type. Packet reliability results are presented from a network and link perspective, whereas prior work only considered the former. We introduce a novel technique of application-aware radio duty cycling called on-demand spatial TDMA. When compared to the non-cycling ...

8 Papers: Wireless data communications using DECT air interface

António Muchaxo, Alexandre Sousa, Nuno Pereira, Helena Sarmento

April 1999 **ACM SIGCOMM Computer Communication Review**, Volume 29 Issue 2

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, references, citings

DECT is an approved ETSI standard for cordless communications, defined as a general radio access technology that can be used as the air interface to any network. In addition to the well-established voice service, it supports data communications. DECT currently addresses low bit rates, but additional modulation options have recently been included for high-speed, up to 2Mbps. In this paper, we describe the hardware and software design of an entire wireless communications system to be used in SOHO ...

Results 1 - 8 of 8

PORTAL

US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+flash +memory +initialization +Vcc

SEARCH

Nothing Found

Your search for **+flash +memory +initialization +Vcc** did not return any results.

You may want to try an Advanced Search for additional options.

Please review the Quick Tips below or for more information see the Search Tips.

Quick Tips

- Enter your search terms in lower case with a space between the terms.

  ```
  sales offices
  ```

  You can also enter a full question or concept in plain language.

  ```
  Where are the sales offices?
  ```

- Capitalize proper nouns to search for specific people, places, or products.

  ```
  John Colter, Netscape Navigator
  ```

- Enclose a phrase in double quotes to search for that exact phrase.

  ```
  "museum of natural history" "museum of modern art"
  ```

- Narrow your searches by using a + if a search term must appear on a page.

  ```
  museum +art
  ```

- Exclude pages by using a - if a search term must not appear on a page.

  ```
  museum -Paris
  ```

Combine these techniques to create a specific search query. The better your description of the information you want, the more relevant your results will be.

```
museum +"natural history" dinosaur -Chicago
```

Terms of Usage Privacy Policy Code of Ethics Contact Us

PORTAL

US Patent & Trademark Office

Subscribe (Full Service) Register (Limited Service, Free) Login

Search: ● The ACM Digital Library ○ The Guide

+flash +memory +initialization SEARCH

THE ACM DIGITAL LIBRARY

Feedback Report a problem Satisfaction survey

Terms used **flash memory initialization** Found **239** of **147,060**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

□ Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

21 Soft updates: a solution to the metadata update problem in file systems

Gregory R. Ganger, Marshall Kirk McKusick, Craig A. N. Soules, Yale N. Patt

May 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 2

Full text available: pdf(147.90 KB) Additional Information: full citation, abstract, references, citings, index terms

Metadata updates, such as file creation and block allocation, have consistently been identified as a source of performance, integrity, security, and availability problems for file systems. Soft updates is an implementation technique for low-cost sequencing of fine-grained updates to write-back cache blocks. Using soft updates to track and enforce metadata update dependencies, a file system can safely use delayed writes for almost all file operations. This article describes soft ...

22 Increasing cache port efficiency for dynamic superscalar microprocessors

Kenneth M. Wilson, Kunle Olukotun, Mendel Rosenblum

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture**, Volume 24 Issue 2

Full text available: pdf(1.09 MB) Additional Information: full citation, abstract, references, citings, index terms

The memory bandwidth demands of modern microprocessors require the use of a multi-ported cache to achieve peak performance. However, multi-ported caches are costly to implement. In this paper we propose techniques for improving the bandwidth of a single cache port by using additional buffering in the processor, and by taking maximum advantage of a wider cache port. We evaluate these techniques using realistic applications that include the operating system. Our techniques using a single-ported ca ...

23 Firmware factory & forth

Brad Eckert

December 1999 **ACM SIGPLAN Notices**, Volume 34 Issue 12

Full text available: pdf(373.72 KB) Additional Information: full citation, citings, index terms

24 Architecture: Leveraging cache coherence in active memory systems

Daehyun Kim, Mainak Chaudhuri, Mark Heinrich

June 2002 **Proceedings of the 16th international conference on Supercomputing**

Full text available: Additional Information:

pdf(217.27 KB) full citation, abstract, references, index terms

Active memory systems help processors overcome the memory wall when applications exhibit poor cache behavior. They consist of either active memory elements that perform data parallel computations in the memory system itself, or an active memory controller that supports address re-mapping techniques that improve data locality. Both active memory approaches create coherence problems---even on uniprocessor systems---since there are either additional processors operating on the data directly, or the ...

Keywords: active memory, address re-mapping, cache coherence

25 Load balanced parallel radix sort
Andrew Sohn, Yuetsu Kodama
July 1998 **Proceedings of the 12th international conference on Supercomputing**
Full text available: pdf(1.22 MB) Additional Information: full citation, references, citings, index terms

26 Location-awareness and interworking: Proximity services supporting network virtual memory in mobile devices
Emanuele Lattanzi, Andrea Acquaviva, Alessandro Bogliolo
October 2004 **Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots**
Full text available: pdf(192.89 KB) Additional Information: full citation, abstract, references, index terms

Wireless networked embedded terminals like personal digital assistants, cell-phones or sensor nodes are typically memory constrained devices. This limitation prevents the development of applications that require a large amount of run-time memory space. In a wired cum wireless scenario, a potentially unlimited amount of virtual memory can be found on remote servers installed on the wired network. However, virtual memory access requires performance constrained and lossless data flows against te ...

Keywords: mobility management, network swapping, proximity service, wireless networks

27 Cellular Disco: resource management using virtual clusters on shared-memory multiprocessors
Kinshuk Govil, Dan Teodosiu, Yongqiang Huang, Mendel Rosenblum
December 1999 **ACM SIGOPS Operating Systems Review , Proceedings of the seventeenth ACM symposium on Operating systems principles**, Volume 33 Issue 5
Full text available: pdf(1.93 MB) Additional Information: full citation, abstract, references, citings, index terms

Despite the fact that large-scale shared-memory multiprocessors have been commercially available for several years, system software that fully utilizes all their features is still not available, mostly due to the complexity and cost of making the required changes to the operating system. A recently proposed approach, called Disco, substantially reduces this development cost by using a virtual machine monitor that leverages the existing operating system technology.In this paper we present a syste ...

28 Where is time spent in message-passing and shared-memory programs?
Satish Chandra, James R. Larus, Anne Rogers
November 1994 **Proceedings of the sixth international conference on Architectural support for programming languages and operating systems**, Volume 29 , 28 Issue 11 , 5

Full text available: pdf(1.55 MB) Additional Information: full citation, abstract, references, citings, index terms

Message passing and shared memory are two techniques parallel programs use for coordination and communication. This paper studies the strengths and weaknesses of these two mechanisms by comparing equivalent, well-written message-passing and shared-memory programs running on similar hardware. To ensure that our measurements are comparable, we produced two carefully tuned versions of each program and measured them on closely-related simulators of a message-passing and a shared-memory machine, ...

29 Cellular disco: resource management using virtual clusters on shared-memory multiprocessors

Kinshuk Govil, Dan Teodosiu, Yongqiang Huang, Mendel Rosenblum

August 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 3

Full text available: pdf(287.05 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Despite the fact that large-scale shared-memory multiprocessors have been commercially available for several years, system software that fully utilizes all their features is still not available, mostly due to the complexity and cost of making the required changes to the operating system. A recently proposed approach, called Disco, substantially reduces this development cost by using a virtual machine monitor that laverages the existing operating system technology. In this paper we present a ...

Keywords: fault containment, resource managment, scalable multiprocessors, virtual machines

30 Integrating performance monitoring and communication in parallel computers

Margaret Martonosi, David Ofelt, Mark Heinrich

May 1996 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1996 ACM SIGMETRICS international conference on Measurement and modeling of computer systems**, Volume 24 Issue 1

Full text available: pdf(1.49 MB) Additional Information: full citation, abstract, references, citings, index terms

A large and increasing gap exists between processor and memory speeds in scalable cache-coherent multiprocessors. To cope with this situation, programmers and compiler writers must increasingly be aware of the memory hierarchy as they implement software. Tools to support memory performance tuning have, however, been hobbled by the fact that it is difficult to observe the caching behavior of a running program. Little hardware support exists specifically for observing caching behavior; furthermore ...

31 Impala: a middleware system for managing autonomic, parallel sensor systems

Ting Liu, Margaret Martonosi

June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 38 Issue 10

Full text available: pdf(684.33 KB) Additional Information: full citation, abstract, references, citings, index terms

Sensor networks are long-running computer systems with many sensing/compute nodes working to gather information about their environment, process and fuse that information, and in some cases, actuate control mechanisms in response. Like traditional parallel systems, communication between nodes is of fundamental importance, but is typically accomplished via wireless transceivers. One further key attribute of sensor networks is that they are almost always long running systems, intended to operate i ...

Keywords: middleware system, sensor networks, software adaptation, software update

32 Disco: running commodity operating systems on scalable multiprocessors
Edouard Bugnion, Scott Devine, Kinshuk Govil, Mendel Rosenblum
November 1997 **ACM Transactions on Computer Systems (TOCS)**, Volume 15 Issue 4
Full text available: pdf(400.76 KB) Additional Information: full citation, abstract, references, citings, index terms, review

In this article we examine the problem of extending modern operating systems to run efficiently on large-scale shared-memory multiprocessors without a large implementation effort. Our approach brings back an idea popular in the 1970s: virtual machine monitors. We use virtual machines to run multiple commodity operating systems on a scalable multiprocessor. This solution addresses many of the challenges facing the system software for these machines. We demonstrate our approach with a prototy ...

Keywords: scalable multiprocessors, virtual machines

33 Integrating reliable memory in databases
Wee Teck Ng, Peter M. Chen
August 1998 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 7 Issue 3
Full text available: pdf(123.18 KB) Additional Information: full citation, abstract, index terms

Recent results in the Rio project at the University of Michigan show that it is possible to create an area of main memory that is as safe as disk from operating system crashes. This paper explores how to integrate the reliable memory provided by the Rio file cache into a database system. Prior studies have analyzed the performance benefits of reliable memory; we focus instead on how different designs affect reliability. We propose three designs for integrating reliable memory into databases: non ...

Keywords: Main memory database system (MMDB), Recovery, Reliability

34 Dynamic self-invalidation: reducing coherence overhead in shared-memory multiprocessors
Alvin R. Lebeck, David A. Wood
May 1995 **ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual international symposium on Computer architecture**, Volume 23 Issue 2
Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper introduces dynamic self-invalidation (DSI), a new technique for reducing cache coherence overhead in shared-memory multiprocessors. DSI eliminates invalidation messages by having a processor automatically invalidate its local copy of a cache block before a conflicting access by another processor. Eliminating invalidation overhead is particularly important under sequential consistency, where the latency of invalidating outstanding copies can increase a program's critical path.DSI is ap ...

35 Hardware/Software Co-testing of Embedded Memories in Complex SOCs
Bai Hong Fang, Qiang Xu, Nicola Nicolici
November 2003 **Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design**
Full text available: pdf(145.29 KB) Additional Information: full citation, abstract

A novel approach for testing embedded memories in complexsystems-on-a-chip (SOCs) is presented. The proposedsolution aims to balance the usage of the existing on-chipresources and dedicated design for test (DFT) hardwaresuch that the functional power constraints are not exceededduring test while trading-off the testing time againstDFT area and performance overhead. The suitability ofsoftware-centric and hardware-centric approaches for embeddedmemory testing is examined and to combine the advanta ...

36 Tempest and typhoon: user-level shared memory
S. K. Reinhardt, J. R. Larus, D. A. Wood
April 1994 **ACM SIGARCH Computer Architecture News , Proceedings of the 21ST annual international symposium on Computer architecture**, Volume 22 Issue 2
Full text available: pdf(1.44 MB) Additional Information: full citation, abstract, references, citings, index terms

Future parallel computers must efficiently execute not only hand-coded applications but also programs written in high-level, parallel programming languages. Today's machines limit these programs to a single communication paradigm, either message-passing or shared-memory, which results in uneven performance. This paper addresses this problem by defining an interface, *Tempest*, that exposes low-level communication and memory-system mechanisms so programmers and compilers can customize polici ...

37 Disco: running commodity operating systems on scalable multiprocessors
Edouard Bugnion, Scott Devine, Mendel Rosenblum
October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles**, Volume 31 Issue 5
Full text available: pdf(2.30 MB) Additional Information: full citation, references, citings, index terms

38 Application restructuring and performance portability on shared virtual memory and hardware-coherent multiprocessors
Dongming Jiang, Hongzhang Shan, Jaswinder Pal Singh
June 1997 **ACM SIGPLAN Notices , Proceedings of the sixth ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 32 Issue 7
Full text available: pdf(1.59 MB) Additional Information: full citation, abstract, references, citings, index terms

The performance portability of parallel programs across a wide range of emerging coherent shared address space systems is not well understood. Programs that run well on efficient, hardware cache-coherent systems often do not perform well on less optimal or more commodity-based communication architectures. This paper studies this issue of performance portability, with the commodity communication architecture of interest being page-grained shared virtual memory. We begin with applications that per ...

39 Performance experiences on Sun's Wildfire prototype
Lisa Noordergraaf, Ruud van der Pas
January 1999 **Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM)**
Full text available: pdf(248.16 KB) Additional Information: full citation, references, citings, index terms

40 Ace: a language for parallel programming with customizable protocols
Mukund Raghavachari, Anne Rogers
August 1999 **ACM Transactions on Computer Systems (TOCS)**, Volume 17 Issue 3
Full text available: pdf(297.50 KB) Additional Information: full citation, abstract, references, index terms, review

Customizing the protocols that manage accesses to different data structures within an application can improve the performance of software shared-memory programs substantially. Existing systems for using customizable protocols are hard to use directly because the mechanisms they provide for manipulating protocols are low-level ones. This article is an in-depth study of the issues involved in providing language support for application-specific protocols. We describe the design and implementat ...

Keywords: parallel processing

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next

... **flash** EPROM front end programmer. ... displaying and writing **memory**, reset function,...
... TELNET, TFTP, HTTP server, asynchronous and HDLC **synchronous** drivers support ...
www.interfaceconcept.com/icserpmcb_html.htm - 34k - Cached - Similar pages
[More results from www.interfaceconcept.com]

[PDF] CE .NET BSP and Service Package for Sharp BlueStreak
File Format: PDF/Adobe Acrobat - View as HTML
... static RAM • 8MB 100 MHz **Synchronous Flash**, 4MB Boot ... 10 Mbps Ethernet Interface • Compact **flash** slot • 2 ... SDRAM controller, interfaces to **memory** media, a ...
www.bsquare.com/dld-files/ datasheets/sharpbsp-ds-1003-5.pdf - Similar pages

[PDF] m in ary Pr elim ina ym inary Pr elim ina y Pr eliminary Pr elim ...
File Format: PDF/Adobe Acrobat - View as HTML
... MB high-speed static RAM, 8 MB 100 MHz **Synchronous Flash**, 4 MB ... Download to RAM or **Flash**. ... system boot support: Reset vector core and **memory initialization**, O/S ...
www.hyubwoo.com/download_data/ bsquare/development/sharp/sharp3_0bsp.pdf - Similar pages

Goooooooooogle ▶
Result Page: **1** 2 3 4 5 6 7 8 9 10 **Next**

Free! Google Desktop Search: Search your own computer.

synchronous flash memory initializat Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2004 Google

Google™

Web Images Groups^New! News Froogle **more »**

synchronous flash memory initialization | Search | Advanced Search
Preferences

Web Results **1** - **10** of about **64,100** for **synchronous flash memory initialization**. (**0.45** seconds)

[PDF] SyncFlash™ **Memory** – A High Speed **Synchronous Flash** Device
File Format: PDF/Adobe Acrobat - View as HTML
... The multi-level (ML) cell **flash memory** burst device ... tu re versions of SyncFlash **memory** will increase ... The **synchronous** burst device's achievable bandwidth is ...
www.semiconductorfabtech.com/datatech/
journals/edition6/downloads/dt6_23_27.pdf - Similar pages

[PDF] 3 Volt **Synchronous** Intel StrataFlash **Memory** to Intel PXA250 ...
File Format: PDF/Adobe Acrobat - View as HTML
... nSDCAS SDRAM Column Address Strobe. Active-low,nSDCAS is used to latch the **initial**- access address during burst reads from **synchronous flash memory**. ...
www.intel.com/design/flcomp/applnots/25069802.pdf - Similar pages

[PDF] Using Intel **Flash Memory**: Asynchronous Page Mode and **Synchronous** ...
File Format: PDF/Adobe Acrobat - View as HTML
... 001 **Initial** Release Page 5. AP-793 5 1.0 Introduction Intel ® **flash memory** products feature two high-performance read modes, asynchronous page and **synchronous** ...
www.intel.com/design/flcomp/applnots/30078301.pdf - Similar pages
[More results from www.intel.com]

Synchronous Flash Memory
... SHARP's **Synchronous Flash Memory** delivers **initial** access times as fast as 60ns and operating frequencies ranging from 40Mhz to 66Mhz. ...
www.sharpsma.com/sma/products/ **memory**/SynchronousFlashMemory.htm - 23k - Dec 9, 2004 - Cached - Similar pages

[PDF] Increase Reliability, Conserve Space BURST MODE **FLASH MEMORY**
File Format: PDF/Adobe Acrobat - View as HTML
... A **synchronous** interface increases performance by looking up ... edge data delivery, and **initial** access Burst ... the code can be executed directly from **Flash memory**. ...
www.amd.com/us-en/assets/content_type/ DownloadableAssets/22237A-BurstMode.pdf - Similar pages

[PDF] AMD Advanced Architecture **Flash Memory** Devices
File Format: PDF/Adobe Acrobat - View as HTML
... while the Am29BDS643 offers an av er age Burst access time of 13.5ns with an **initial synchronous** access time of ... AMD Advanced Architecture **Flash Memory** Devices ...
www.amd.com/us-en/assets/content_type/ DownloadableAssets/nvdadvarch.pdf - Similar pages
[More results from www.amd.com]

Asynchronous **synchronous** VME PowerQUICC II PowerPC MPC8250 HDLC ...
... and timer management, **memory** management, mapping of **memory** spaces, basic ... **Flash** disk based on 8 bits **Flash** 64, 128 ... Four Asynchronous/**Synchronous** Serial ports. ...
www.interfaceconcept.com/plinesevmea_html.htm - 42k - Cached - Similar pages

Asynchronous **synchronous** PMC board Ethernet port HDLC SDLC ...